Autonomic Computing via Dynamic Self-Repair of Hardware Faults

Second Annual Report: January 1, 2005 - January 1, 2007

PI Daniel J. Sorin, co-I Sule Ozev

Department of Electrical and Computer Engineering

Duke University

1 Introduction

The goal of this research project is to develop computer systems that use dynamic self-repair to enable *autonomic operation in the presence of permanent hardware faults*. Autonomic operation is crucial for reliability when mission-critical computer systems are deployed for a long time with little or no opportunity for human repair, as in NASA's mission to Mars or other long-duration missions. Autonomic operation is also becoming more important for commercial computer systems, as both the number of computers in an installation and the hardware fault rates continue to rise. Autonomic systems detect errors, diagnose their underlying faults, and take action to mask hard faults. In this research proposal, we focus on microprocessors, since they comprise the majority of computational infrastructure, although we plan to apply our techniques more broadly (e.g., to network processors, digital controllers, etc.).

As microprocessor fabrication technology continues to shrink devices and wires and increase clock frequencies, hard fault rates in hardware are consequently increasing. For example, with decreasing feature sizes, the mean time to breakdown for interconnects due to electromigration decreases [10], making hard faults more likely. As another example, shrinking transistor gate oxide thickness also decreases the lifetime of transistors under continuous operational stress [11, 9]. These effects are exacerbated by stressful operating conditions, such as those experienced by computers that operate outside of the protection of Earth's atmosphere. Moreover, with increasing numbers of transistors being used, the probabilities of microprocessor hard faults are correspondingly increasing.

Existing solutions for tolerating hard faults are either very expensive—in terms of power and hardware cost—or suffer unacceptable performance penalties for many classes of hard faults. One class of approaches uses redundant parallel processors (e.g. triple modular redundancy) to provide forward error recovery (FER). These systems provide high reliability and performance, but they use large amounts of power and hardware. For NASA, in particular, this extra power consumption is generally not acceptable. At the other end of the spectrum, a recently developed backward error recovery (BER) scheme, called DIVA [2], uses only a small on-chip checker to achieve almost as much reliability as redundant processor schemes. DIVA uses much less hardware and power, but it incurs significant performance and energy pen-

alties for recovery every time a fault is exercised. These penalties can be particularly problematic for hard faults in heavily used circuits, such as the reorder buffer.

In this research project, we seek to develop lightweight hardware techniques for self-repair that can achieve high reliability and robust performance without consuming vast amounts of power or hardware. In the first two years of this project, we have made more progress than we had expected, and we are very excited about the achievements to date and the research that we will perform in the third year.

2 Achievements to Date and Plans for Continuing Research

In this section, we discuss what we have achieved thus far and what we plan to continue working on. We present this in the framework of the research plan that was in our original proposal (original plan in *italics*).

- *We will improve our initial implementation of Self-Repairing Array Structures (SRAS) and evaluate its potential to protect a variety of array structures within modern microprocessors.*

We have largely achieved this goal. First, we extended SRAS [6] by creating a second implementation of it that uses error detecting codes (EDC) instead of a more complicated approach for detecting errors. We have published this work in IEEE Transactions on Dependable and Secure Computing [5].

Second, we developed an integrated approach to detecting, diagnosing, and reconfiguring around hard faults in general microprocessor structures, not just arrays. This research was published at the highly selective (20% acceptance rate) 2005 International Symposium on Microarchitecture (MICRO) [7], and it was presented at the conference by a graduate student in our research group. The key to our scheme is the diagnosis mechanism. As each instruction traverses the microprocessor pipeline, it records which structures it has used (e.g., ALU#2, reorder buffer entry #23, and load/store queue entry #17). If the instruction is detected to be faulty, the error counters for each of these components is incremented. If any of the counters exceeds its threshold, it indicates a hard fault, since transient faults will never lead to above-threshold counters (because we clear the counters periodically). In conjunction with this diagnosis mechanism, we use an existing technique for error detection/correction, DIVA [2], and new and existing techniques for reconfiguring the processor to avoid using faulty components. After publishing this MICRO paper, we extended the experimental analysis of our diagnosis mechanism and this extended paper has been accepted for publication in ACM Transactions on Architecture and Code Optimizations [8].

- *We will develop detailed hard fault models that correspond to the underlying physical phenomena that occur in modern VLSI technology.*

We developed a high-level hard fault model for a 64-bit adder in a microprocessor for use in our MICRO paper, and we have been working to extend this to other structures. On a related topic, we developed a new metric for evaluating a processor's resilience to hard faults. This metric enables us to quantitatively com-

pare different approaches to tolerating hard faults in a given computer structure. Our metric, which we published in ACM SIGMETRICS [4], is based on prior work that developed the Architectural Vulnerability Factor (AVF) that applies only to transient faults [3]. The previous AVF metric provides misleading results if naively applied to hard fault scenarios.

Recently, we have begun to look at modeling faults due to the impact of CMOS process variability. As microprocessors are being fabricated with smaller and smaller transistors, the variability in their dimensions is becoming significant. For example, if each transistor in a 45nm process is a few nanometers shorter or longer than nominal, due to fabrication variability, it can have a non-trivial impact on device, circuit, and processor behavior. We explored the impact of variability on a microprocessor pipeline, and this preliminary research has been accepted for publication at the 2nd Workshop on Architectural Reliability [12]. This paper is, to the best of our knowledge, the first paper to quantify the impact of variability on a complete microprocessor (not just on selected components). We plan to continue this research, because variability threatens the computer industry's ability to reliably and efficiently use future CMOS technologies.

- *We will fully develop and evaluate Hierarchical Modular Redundancy (HMR), including efficient fault detection and remapping schemes.*

We began this research by exploring HMR for n-bit adders and n-bit multipliers, and the initial results are promising. These fault-tolerant adders and multipliers can tolerate more hard faults than existing schemes (including full replication) with less hardware cost and less power consumption. We have begun integrating these designs into the microprocessor simulator to explore their impact on the overall behavior, and we published the results for the multiplier in the 2006 International Test Conference [14]. We plan to perform detailed VLSI layouts of these designs to ensure that they can be fabricated without undue difficulty.

Future work in this area will focus on HMR for floating point units and for the scheduling logic in the microprocessor's dynamically scheduled core.

- *We will apply SRAS and HMR to system models other than just microprocessors, including network processors and embedded controllers.*

This research is mostly still in the future, although we have recently developed a scheme for more efficiently tolerating faults, both transient and permanent, in cache memories. This work was published in the 2006 International Conference on Computer Design [13], and it is applicable to caches in any type of computer system.

- *We will, in conjunction with all of the above plans, develop simulation infrastructure for evaluating our ideas—in terms of power, hardware, and performance—at the architectural, circuit, and device levels.*

We have already developed simulation tools for exploring the impact of hard faults on microprocessors. This aspect of the development has been built on top of the industry-standard SimpleScalar simulator [1]. We are working to develop an automated and parameterized method of injecting hard faults into SimpleScalar. The tool development at lower levels (circuits and devices) has thus far mostly consisted of using software scripts to incorporate existing tools such as SPICE.

We are in the process of developing a custom tool set for exploring the impact of CMOS process variability on circuit and system behavior. This tool set can take a Verilog or VHDL description of a circuit or system and perform a statistical timing analysis on it to determine its performance and reliability. We plan to distribute our tool set to the community, in order to enable other researchers to more easily make contributions in this field.

3 Education and Training

Beyond the obvious research contributions of this project, it has also provided support and infrastructure for the education and training of two graduate students (Mahmut Yilmaz and Bogdan Romanescu) and one undergraduate research assistant (Derek Hower). Derek Hower participated in the research program as part of Duke's prestigious Pratt Research Fellow program, before graduating and pursuing graduate research at the University of Wisconsin. We will begin working with a new undergraduate Pratt Fellow, Michael Bauer, in Spring 2007. Throughout this project, we have worked with Fred Bower, who is a full-time IBM employee and part-time graduate student. He is funded by IBM.

References

[1] T. Austin, E. Larson, and D. Ernst. SimpleScalar: An Infrastructure for Computer System Modeling. *IEEE Computer*, 35(2):59–67, Feb. 2002.

[2] T. M. Austin. DIVA: A Reliable Substrate for Deep Submicron Microarchitecture Design. In *Proc. of the 32nd Annual International Symposium on Microarchitecture*, pages 196–207, Nov. 1999.

[3] A. Biswas et al. Computing Architectural Vulnerability Factors for Address-Based Structures. In *Proceedings of the 32nd Annual International Symposium on Computer Architecture*, June 2005.

[4] F. A. Bower, D. R. Hower, M. Yilmaz, D. J. Sorin, and S. Ozev. Applying Architectural Vulnerability Analysis to Hard Faults in the Microprocessor. In *Proceedings of the 2006 ACM Sigmetrics Conference on Measurement and Modeling of Computer Systems*, June 2006.

[5] F. A. Bower, S. Ozev, and D. J. Sorin. Autonomic Microprocessor Execution via Self-Repairing Arrays. *IEEE Transactions on Dependable and Secure Computing*, 2(4):297–310, Oct-Dec. 2005.

[6] F. A. Bower, P. G. Shealy, S. Ozev, and D. J. Sorin. Tolerating Hard Faults in Microprocessor Array Structures. In *Proceedings of the International Conference on Dependable Systems and Networks*, pages 51–60, June 2004.

[7] F. A. Bower, D. J. Sorin, and S. Ozev. A Mechanism for Online Diagnosis of Hard Faults in Microprocessors. In *Proceedings of the 38th Annual IEEE/ACM International Symposium on Microarchitecture*, Nov. 2005.

[8] F. A. Bower, D. J. Sorin, and S. Ozev. Online Diagnosis of Hard Faults in Microprocessors. *ACM Transactions on Architecture and Code Optimization*, To Appear 2007.

[9] K. C. Boyko and D. L. Gerlach. Time Dependent Dielectric Breakdown at 210 Å Oxides. In *Proceedings of the 27th Annual Reliability Physics Symposium*, pages 1–8, Apr. 1989.

[10] Z. Li, G. Wu, Y. Wang, Z. Li, and Y. Sun. Numerical Calculation of Electromigration Under Pulse Current with Joule Heating. *IEEE Transactions on Electron Devices*, 46(1):70–77, Jan. 1999.

[11] B. P. Linder et al. Growth and Scaling of Oxide Conduction After Breakdown. In *41st Annual IEEE International Reliability Physics Symposium Proceedings*, pages 402–405, Mar. 2003.

[12] B. F. Romanescu, S. Ozev, and D. J. Sorin. Quantifying the Impact of Process Variability on Microprocessor Behavior. In *Proc. of the 2nd Workshop on Architectural Reliability*, Dec. 2006.

[13] N. N. Sadler and D. J. Sorin. Choosing an Error Protection Scheme for a Microprocessor's L1 Data Cache. In *Proceedings of the International Conference on Computer Design*, Oct. 2006.

[14] M. Yilmaz, D. R. Hower, S. Ozev, and D. J. Sorin. Self-Detecting and Self-Diagnosing 32-bit Microprocessor Multiplier. In *International Test Conference*, Oct. 2006.